A Parallel Toolset for Intervisibility Analysis

Tom KREITZBERG
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA USA

Abstract. The ability to determine line-of-sight over a digital elevation model (DEM) is useful in a variety of fields. Since line-of-sight and related problems involve computations on a grid with radial data dependence, domain decomposition on a distributed memory parallel machine is a good approach to solving them in a timely manner. This paper describes a set of intervisibility tools offering the analyst a) viewshed determination for a single reference point; b) Boolean operations on viewshed masks for a set of reference points; c) statistical matrices for the complete intervisibility problem (the set of all viewsheds) for a DEM; and d) radio path loss computation. The toolset has been implemented on a 16-node transputer hypercube running Express, and is available to the analyst via an X-based graphical user interface.

ntroduction

The capability to rapidly determine which points on a given Digital Elevation Model (DEM) are visible from a fixed reference point is important in areas as diverse as land management, radio communications, and military planning [1]. All such problems in the class of intervisibility analysis share a property which may be called *radial data dependence*: the output at a point depends in a fundamental way on the input data along a ray passing through that point.

This property of regular data dependence suggests that intervisibility analysis may be efficiently performed using data decomposition on a distributed memory parallel system. Such a decomposition has already been shown to work for perspective view rendering, another problem with radial data dependence [2]. A set of intervisibility tools which use the same paradigm has been developed at the Jet Propulsion Laboratory (JPL). The toolset, which addresses a large variety of problems -- from simply determining whether one hilltop is visible from another, to complex problems of optimal antennae placement -- has been implemented on a network of sixteen transputers on a Sun host, using the Logical Systems C compiler within the Express transputer environment.

The user interface is handled through a menu-driven map background package, running on a (possibly remote) X workstation. Tools include the following:

1) Viewshed determination for a single reference point; *viewshed* is the term for the set of points visible from the reference point. The product is a Boolean mask of hidden/visible DEM points.

2) Boolean operations on viewshed masks for a set of reference points. The products are themselves Boolean images referring to points visible to all reference points ("AND"); points visible to at least one reference point ("OR"); and so forth. Any combination of Boolean operations on the viewsheds of reference points may be constructed.

3) Generation of statistical matrices for the viewsheds of all the points in the reference DEM. Statistics include size of total viewshed, size of connected viewshed (the set of points visible to the reference point with no hidden points in between), and viewshed size gradient.

4) Radio path loss computation, a matrix of the power drop-off of a radio signal due to distance, terrain, and scattering; and the corresponding signal-to-noise ratio at all DEM points for a given transmitter signal strength.

The products of all the tools may be displayed against a geo-referenced map background, or used as input to more complex analytical tools, such as optimal path planning and resource allocation analysis.

Section 2 of this paper describes the method of tiled radial domain decomposition used to parallelize the elements of the toolkit. Section 3 is a discussion of some memory and I/O considerations. Section 4 describes the user interface and analyst options, and Section 5 concludes the paper with directions for future work.

2. Radial Domain Decomposition

2.1 The Basic Decomposition Scheme

The fundamental operation performed by all intervisibility tasks is to compute an output value at a point as a function of input data along a ray which passes through that point. This radial data dependence suggests a natural approach to implementing the toolset on a transputer network: assign each node an independent area of responsibility (AOR), bounded by the edges of the DEM and rays originating at the reference point. A node is responsible for computing all output data along the rays contained by its AOR (Figure 1).

This approach offers several benefits when implemented on transputers. Since only the input data covering a node's AOR is required locally, communications overhead and memory requirements are substantially lower compared to methods which ignore the radial data dependence. An arbitrary number of nodes may be used, with the number varying from run to run. In fact, if local memory constraints are such that the entire problem does not fit when decomposed across the available nodes, it is a straightforward matter to break the problem up so that each node computes the output for several AORs, with the nodes running in parallel until the complete DEM is finished.

By using three auxiliary arrays, it is possible to efficiently map a wedge-shaped region of a two-dimensional grid into a one-dimensional data array in a C program using the Express library [3]. For an MxN DEM, define

start[i] = j, where element (i,j) is the first element in row i used by the node;
wide[i] = k, where k is number of elements in row i used by the node; and
offset[i] = $(\sum_{l=0}^{i-1} \text{wide}[l])$ - start[i]; for $0 \le i < M$.

If no elements in row i are used, let start[i] = 0. If the input data is stored in a one-dimensional array *data* beginning with row 0, the C macro

```
#define ELEMENT(line, sample) data[offset[line]+sample
```

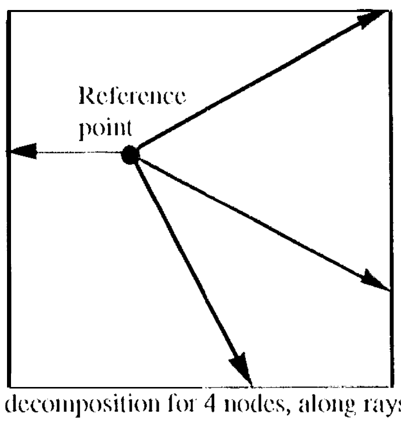

Figure 1. Equal area domain decomposition for 4 nodes, along rays from the reference point to the boundary of the DEM.

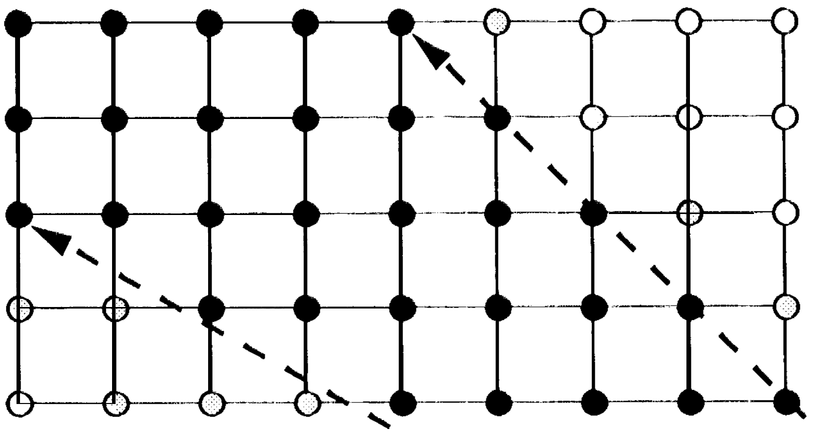

Figure 2. A portion of a node's area of responsibility (AOR). The dashed lines are the bounding rays from the reference point. The solid gridpoints are those for which the node supplies the output values. Input data is required at the solid as well as the shaded gridpoints, to interpolate elevation values along the bounding rays.

returns the element with global coordinates (line, sample). The macro

```
#define VALID(line, sample) ((sample >= start[ line ]) && (sample<start[line] +\
                                                  wide[line]))
```

returns 1 when the element (line, sample) is actually stored in *data*.

Given these macros, the C code for generating the output using radial domain decomposition is identical to the version using no decomposition, except for the substitution of ELEMENT(line,sample) for data[line][sample] (and, for the timid, appropriate use of VALID).

The auxiliary arrays *start* and *wide* represent the set of data required by a node. In general, and in the particular case of line-of-sight, the sets of points needed for input and output will not be the same. For output only those points on or within the AOR boundary are required, but to compute elevation values along the bounding rays, values at points adjacent but external to the boundary are used (Figure 2). Therefore two sets of auxiliary arrays, one for input and one for output, are computed.

2.2 Tiled Radial Domain Decomposition

Radial domain decomposition is a reasonable first step to parallelizing intervisibility problems, but it does not adequately deal with the relatively high cost of I/O operations. The row-based scheme implies that on the order of M reads, each on the order of N bytes, from the host are necessary to load the DEM data. The host could concatenate the data, but the difficulty remains of moving the data between nodes as subsequent reference points, creating new radial decompositions, are received.

One solution is to reduce the granularity of the input data decomposition. The input data may be considered transmittable only in units of tiles or blocks, usually square sets of contiguous gridpoints (Figure 3). Rather than optimizing memory usage by requesting precisely those data the node will use to compute the output, the node requests the set of tiles covering the necessary gridpoints. If the data are in square tiles m points across, the number of reads is reduced to O(M/m), with each read involving O(N*m) bytes.

The auxiliary arrays *start* and *wide* become faster to compute, having shrunk in length to at most ⌈M/m⌉ + 2 (the exact length depends on in which row of the tiles the first row of the DEM falls). Since overall difficulty, like entropy, seems always to be a non-decreasing function, computing the offset array is much harder. In fact, storing the data as tiles locally makes two offset arrays, of dimensions M and N respectively, useful.

Let the DEM element (0,0) be element (o,p) in the m-by-m tile containing it. Then DEM row i falls in tile row q=(i+o)/m, and DEM column j is in tile column r=(j+p)/n. If DEM element (i,j) is stored in a node's data array, there are

$$(\sum_{l=0}^{q-1} wide[l]) + (r - start[q])$$

tiles stored before tile (q,r); the summation counts the tiles in rows above the element, and the rest of the expression counts the tiles stored in that tile row before tile (q,r). DEM element (i,j) is element

$$((i+o)\%m)*n + ((j+p)\%n)$$

in tile (q,r); here "%" is used for the remainder operator.

Now, define the arrays *roffset* and *coffset* as follows:

$$roffset[i] = (\sum_{l=0}^{[(i+o)/m]-1} wide[l] - start[[(i+o)/m]])*m*n + ((i+o)\%m)*n,\ 0 \le i < M;$$
$$coffset[j] = [(j+p)/n]*m*n + (j+p)\%n,\ 0 \le j < N.$$

(Operations within square brackets return integer values.) Then, assuming DEM element (line, sample) is stored locally, the C macro

```
#define ELEMENT(line, sample)  data[roffset[line] + coffset[sample]]
```

maps the global coordinates into the local array. The macro VALID, for determining whether a global data point is in fact stored locally, is more complicated than the version given in the previous section. One way of expressing it is

```
#define TILE_ROW(line)  (line+o)/m
#define TILE_COLUMN(sample) (sample+p)/n
#define VALID(line, sample) ((TILE_COLUMN(sample)>= start[TILE_ROW(line)]) \
 && (TILE_COLUMN(sample) < start[TILE_ROW(line)] + wide[TILE_ROW(line)]))
```

This is attractive neither aesthetically nor computationally, but a carefully written program should not need to use it.

3. Memory Considerations and Data Exchange

The assumed scenario for the intervisibility toolkit is one in which the analyst selects an area of interest and requests a series of tasks involving a series of reference points. This

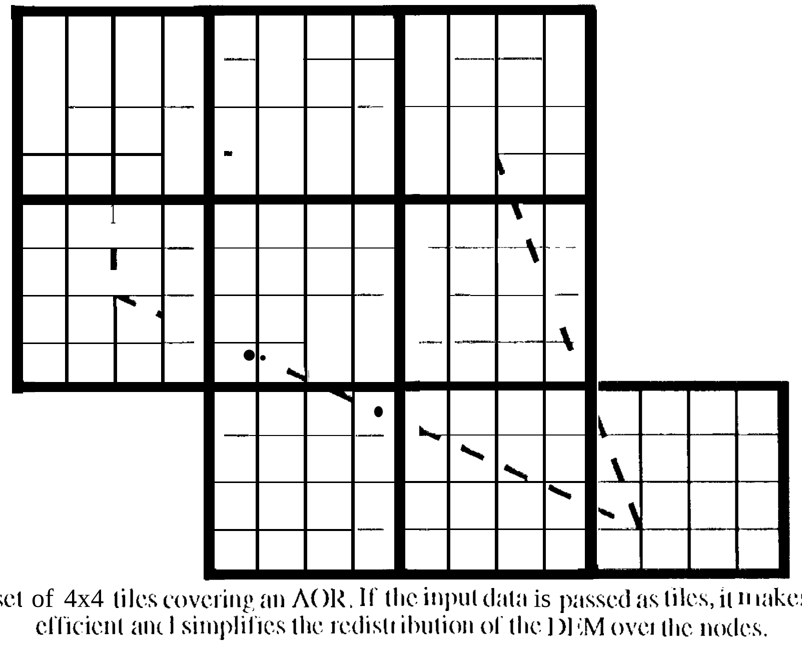

Figure 3. The set of 4x4 tiles covering an AOR. If the input data is passed as tiles, it makes I/O calls more efficient and simplifies the redistribution of the DEM over the nodes.

produces two problems regarding DEM data movement. The first is loading the DEM data from the host into the transputers in a time- and memory-efficient manner; the second is rearranging the DEM as required for new reference points and new domain decompositions.

3. Tiled Tagged Image ile ormat

Tiled radial domain decomposition is employed to address the first data movement problem. The advantages for the transputers have already been discussed, but tiling is also useful for the host computer and the human data manager.

The Tagged Image File Format (TIFF[1]) is a tag-based file format maintained by Aldus Corporation [4]. TIFF Revision 6.0 supports tiled images, and JPL has developed TIFF 6.0 compliant software for adding geographic information to the files [5]. The elevation data, therefore, may be stored and accessed as a tiled TIFF image by the host machine.

There are several advantages to using tiled TIFF elevation files. TIFF is a well-maintained, standardized format which is gaining in popularity, and a C language source code of routines for creating and displaying TIFF images is available in the public domain. TIFF supports a number of data compression schemes, and the files are directly transportable across a variety of platforms, without concern for big-endian/little-endian and byte-swapping differences; these features address several concerns for the person responsible for generating and maintaining the data files. Finally, accessing a subregion of a large tiled image is a much more efficient process than accessing an image stored by row.

3.2 The Hardware Platform, Problem Sizes, and ata Shuf ling

The target platform for the intervisibility toolset is a network of T805 transputers in a 16-node hypercube configuration, connected to the VME bus of a Sun SparcServer 670 MP (Figure 4). Four of the sixteen nodes have 16 Mb of local memory; the others each have 4 Mb. Intervisibility tasks are not guaranteed the entire hypercube, but will be allocated nodes in sub-cubes of at least 4.

For a given subset of this network and a fixed DEM, there are a number of possible combinations of required to available memory. In the simplest case, the entire problem -- input DEM, output, other data, and code -- takes less than 4 Mb, and therefore can fit on each node. Assuming a 4 byte DEM and a 1 byte output, grids of around 800 by 800 should fit on a 4 Mb node, in which case no radial distribution of the input data is necessary.

A second possibility is that the memory requirements are between 4 and 16 Mb. This means that some sort of decomposition is required, but the entire DEM will fit on one of the four "corner" 16 Mb nodes. When the DEM is first loaded from the host, a corner node stores all the data, while the other nodes receive only those tiles which cover their AORs. When a subsequent request arrives, requiring a new domain decomposition, the 4-Mb nodes determine which new tiles are required and which current tiles may be discarded; the new tiles are requested from the corner node, which does not need to compute its own tile set. If the process' subcube contains more thai 16-Mb node, ile requests may be distributed among them.

This possibility generalizes to the case where memory requirements are greater than that available to the node with the least memory but less than the total memory in the subcube. A list of which tiles each node is responsible for broadcasting must be maintained, and updated as the tiles migrate throughout the network. If the list is propagated globally, each node will be able to send a single message to every other node, requesting the appropriate tiles. Once a node has received requests from all other nodes (some of which may be null), it may route a tile to all nodes which requested it with a single broadcast message.

[1]TIFF is a trademark of Aldus Corporation

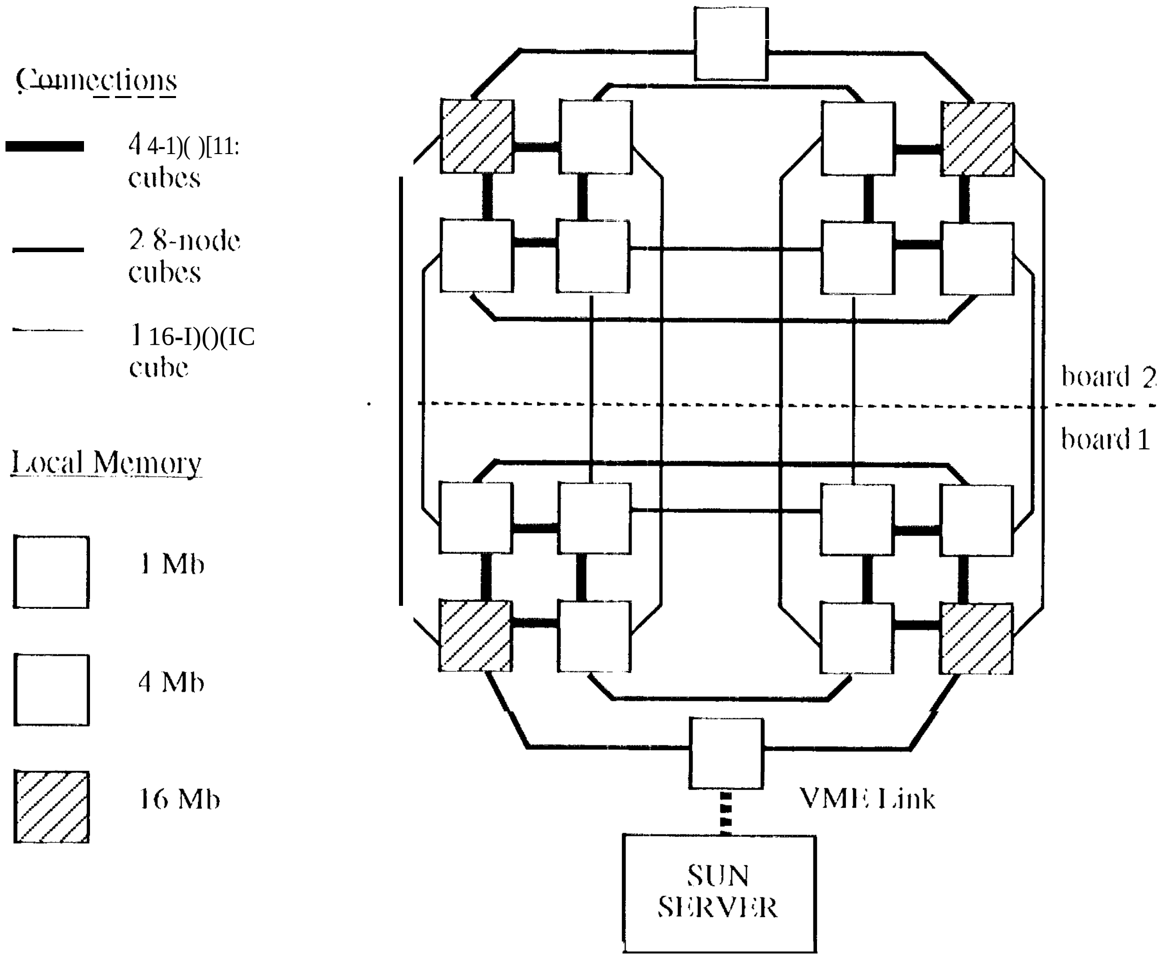

Figure 4. The transputer platform comprises two identical boards; a VME link switch distinguishes the board which communicate with the host. Hypercube message routing efficiency is unchanged for all but the four "corner" 16 Mb nodes; these require an extra node in the routes between pairs in the upper and lower subcubes, but their maximum message distance is still 4 nodes. From controller node, maximum distance is also 4 nodes.

4. User Interface and Options

The intervisibility toolset comprises three principle functions, each of which is fully described below. While the toolset itself was designed to service requests from any source, including other applications running on the transputer network, there is a user interface, developed along with the toolset, which provides simple yet complete access to the capabilities of the toolset, as well as a variety of means for displaying the products. The interface is X-Motif based, and runs under (at least) SunOS 4.1.2.

The user interface operates within the environment of the Mapper Toolkit, a situation display manager implemented at JPL [6]. The Mapper Toolkit provides an analyst with a georeferenced map background, along with a variety of useful functions such as roaming, zooming, an overlay drawing package, symbol placement, coordinate conversion, and distance calculation. The primary map source is Arc Digitized Raster Graphics (ADRG) data from the Defence Mapping Agency (DMA); these maps are shipped on compact discs, then read and converted into tiled TIFF files for ready access and display.

In a typical scenario, the analyst would use the Mapper Toolkit to position a map over the desired Area of Interest (AOI), then select "Visibility Analysis" from the "Applications" pulldown menu.

4.1 Visible Line-of-Sight Viewshed Computation

Visible line-of-sight (LOS) determination is the basic function of the intervisibility toolset. It answers the question, "Which points within the AOI are visible from a fixed reference point?" The reference point is, of course, given in three dimensions: in addition to the geographical coordinates, a height above the ground is required. The reference point need not, in general, be within the AOI, though for most LOS applications it is.

Two points are visible to each other (the operation is obviously symmetric) when the line segment connecting them does not pass through the ground; that is, when it lies entirely above the elevation profile between the two points. Given the ray from a reference point to a visible point, all points beyond the visible point which are beneath that ray may be said to be in the point's "shadow," and therefore not visible. The first point on or above that ray is visible, and casts its own shadow at least as far as all visible points before it. Since the closest point is always visible, a simple pseudo-code algorithm for line-of-sight along an elevation profile is as-follows:

```
LOS[1] = VISIBLE;
Slope = Elevation[1] / Distance[1];
i = 2;
while (i < PROFILE_LENGTH) (
    while (Elevation[i] < Slope * Distance[i]) {
        LOS[i] = HIDDEN;
        i ++;
    }
    LOS[i] = VISIBLE;
    Slope = Elevation[i] / Distance[i];
}
```

(This assumes that the elevation profile has been computed with the height of the reference point set to zero.)

If the analyst is only interested in whether two particular points are visible to each other, the computation is done on the Sun host, since the SPARC processor is faster than a single transputer. If the set of all points in the AOI visible to the reference point -- the *viewshed* of the reference point -- is required, each node executes a version of the above pseudo-code for the elevation profiles from the reference point to each boundary point in its AOR. After the LOS array has been filled, the binary VISIBLE/HIDDEN information is interpolated onto the output grid, with the hidden output pixels set to 0 and the visible ones set to 1. When the entire output grid has been computed, results are collected, compressed from 1 byte per datum to 1 bit per datum, and sent to the host.

The output product of visible LOS is therefore a binary mask, with each point in the output grid assigned a value. This logical data is in a form such that other terrain grid-based reasoning applications, such as asset schedulers and route planners, could easily incorporate it into their reasoning procedures [7]. Or it may simply be displayed as an overlay on a map background, shading the areas visible to the reference point.

Furthermore, a set of these binary masks may be combined using logical bitwise functions, such as AND and OR, to create a composite overlay. Some logical combinations have fairly obvious interpretations: AND produces a mask of points visible to all reference points; OR produces a mask of points visible to at least one reference point. Others are somewhat more abstract; (A AND B) XOR C yields a mask of points only visible to reference points A and B, or to reference point C. The user interface allows arbitrary bitwise logical combinations to be specified for a series of reference points; currently, except for AND and OR, which both have simple English explanations, it is up to the user to provide her own interpretation of the expression she supplies.

4.2 Intervisibility Matrices

The elements of an *intervisibility matrix* (IVM) contain the size of the viewshed at a fixed height above the gridpoint. Such a matrix is useful in land-use planning as well as military sensor placement problems.

To generate an IVM, the transputers must compute the LOS masks at each point on the grid. For small grids (less than about 800 by 800), each node can store the entire input DEM; therefore, the domain decomposition is by strips of the output matrix. A node computes the entire viewshed for each point in a horizontal band of the AOI grid.

Larger grids, however, require radial domain decomposition. All nodes work in parallel to compute the viewshed of each gridpoint, as with a single LOS problem described above. The reference point progresses by input tile, rather than by row. This order will reduce the incremental change in the set of tiles stored locally, since the radial decompositions will change only slightly as the reference point moves about within the tile.

Once the IVM has been computed, a variety of display options remain for the analyst. The data may be displayed as a grayscale map overlay, with the brightest points in the overlay indicating the largest viewsheds; up to seven colors may be used instead of shades of gray, with each color representing an interval of viewshed sizes; and the data may be cut off with lower and upper thresholds. All of these options are available interactively after the matrix has been returned to the process which requested it.

The intervisibility toolset also generates three variations on the IVM: a filtered IVM; the gradient of the IVM; and a connected viewshed IVM. The *filtered IVM* is produced by running a 3x3 box filter (i.e., the output at a gridpoint is the average of the input and its eight surrounding neighbors) on the intervisibility matrix; this filter is performed in parallel, with each node responsible for filtering a horizontal band of the output matrix. Display options are the same as for the full IVM.

The gradient operation produces flow lines through each gridpoint, indicating the direction of the greatest incremental increase in the size of the viewshed. As with the filtered IVM, the gradient is computed with a horizontal strip domain decomposition after the IVM is generated with a radial domain decomposition. The flow lines for each strip are computed in parallel, then connected across strip boundaries; the final product is a set of vectors tracing viewshed sizes from local minima to local maxima.

An element in a *connected viewshed IVM* represents the size of the polygon comprising the set of visible points which have no hidden points between themselves and the reference point; that is, given an elevation profile, the points between the reference point and the first hidden point are precisely those included in the connected viewshed. The connected viewshed IVM is computed in the same manner as the full IVM, except that the first hidden point, rather than the boundary point, is the terminating condition in the loop along an elevation profile. Display options are the same as for full and filtered IVMs.

4.3 Radio Signal Path .oss

For radio signal propagation, visible line-of-sight is neither necessary nor sufficient .o guarantee successful communication. Effects such as knife-edge refraction and tropospheric scattering extend the receivability of a radio signal beyond the visible LOS, while distance and atmospheric effects attenuate the signal [8].

The Department of Defence's Electronic Compatibility Analysis Center (ECAC) has developed the Terrain Integrated Rough Earth Model (TIREM) software package for computing radio signal path loss [9]. TIREM has been converted to run in parallel on the transputer network to service radio LOS requests for the intervisibility toolkit.

As with visible LOS, radio LOS operates along an elevation profile. Radial domain decomposition is therefore used in an identical manner for both problems.

In addition to the reference point, a receiving antenna height and a propagation frequency are required for radio LOS. The analyst may choose to display the raw results of signal path loss, convert the data to signal strength by providing a transmission power, or display only the contours bounding areas above a given signal-to-noise ratio.

5. Conclusions and Future Work

This paper describes the implementation of a suite of intervisibility tools on a Sun-hosted ransputer network. The property of radial data dependence common to all the problems addressed by the toolkit was exploited to minify the memory and communication requirements for solving a series of problems regarding the same input DEM.

A number of questions remain regarding the optimization of this implementation. In particular, detailed analysis of the costs of the various schemes discussed in Section 3.2 for distributing large DEMs over the nodes needs to be performed, to ensure that the extended memory of the 16-Mb transputers is properly used, guarding against both the bottleneck of these nodes being swamped by requests for new tiles, and the cost of maintaining a distributed list if the DEM is scattered over the system in a less systematic manner.

User feedback is required to spur the development of the interactive user interface. The ideas of the developer regarding appropriate capabilities, warped by an incomplete understanding of how the toolset may be used by an expert and by the desire to try the easiest choice first, do not always match the ideas of the end-user.

Finally, the generic interface to the intervisibility toolset needs to be hardened and codified, to make the programs a uly usable source of ntervisibility data available to applications programs in general.

Acknowledgements

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the Director of Advanced Technology, PM-ASAS, through an agreement with the National Aeronautics and Space Administration.

References

[1] K. Clarke, *Analytical and Computer Cartography*, Prentice Hall, Englewood Cliffs, NJ, 1990.
[2] S. Groom and S. Watson, Using the Delta for Solar System Visualization, *NASA OSSO Information Systems Newsletter*, October 1992, pp. 7-12.
[3] T. Kreitzberg, A Method of Radial Domain Decomposition, *World occam and Transputer Users Group Newsletter* **17** (1992) 35-41.
[4] Aldus Developers Desk, TIFF Revision 6.0, Aldus Corporation, Seattle, June 1992.
[5] N. Ritter, TIFF 6.0 Compliant Subroutine Library, JPL Technical Note, June 1992.
[6] J. Skinner *et al.*, The Mapper Toolkit: Situation Display for Geographical Data, ᴾI Technical Note, February, 1992.
[7] D. Sapounas & T. Kreitzberg, The Tactical Movement Analyzer, *Proceedings of the Third Workshop on Battlefield Intelligence in AirLand Operations*, Las Cruces, NM, 1992.
[8] M. Hall, *Effects of the Troposphere on Radio Communication*, Peregrinus, New York, 1979.
[9] R. Sciandra, *TIREM/SEM Programmers Reference Manual*, ECAC-CR-90-039, July 1990.